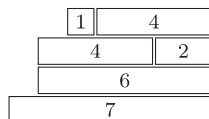


Informatyczny kącik olimpijski (93): Wieża z siana

W tym miesiącu zadanie *Tower of Hay*, które pojawiło się na konkursie USACO Open Gold w roku 2009. Z n prostokątnych bel siana, które mają tę samą wysokość, ale różne szerokości w_1, w_2, \dots, w_n chcemy zbudować wielopoziomą wieżę. Na każdy poziom może się składać kilka bel, a sumaryczna szerokość każdego poziomu musi być nie większa niż sumaryczna szerokość poziomu znajdującego się bezpośrednio pod nim (o ile taki istnieje). Co więcej, żadna bela nie może znajdować się na wyższym poziomie niż inna bela o wyższym numerze (czyli należy je układać po kolei) i należy wykorzystać wszystkie bele. Jaka jest największa możliwa wysokość wieży, którą można zbudować przy takich założeniach (patrz rys. 1)?



Rys. 1. Dla $n = 6$ bel o kolejnych szerokościach 7, 6, 4, 2, 1, 4 można zbudować wieżę o wysokości 4.



Rys. 2. Algorytm zachłanny dla bel z rysunku 1 zbuduje wieżę o wysokości 3.

Dość naturalnym rozwiązaniem zachłannym, które może się nam narzucić, jest konstruowanie każdego poziomu z jak najmniejszej liczby bel siana, podczas budowania wieży od góry. Niestety, jest to rozwiązanie niepoprawne, jak można się przekonać, patrząc na rysunek 2.

Spróbujmy więc rozwiązania opartego o metodę programowania dynamicznego. Niech $d[i, j]$ oznacza maksymalną wysokość wieży złożonej z bel o szerokościach w_i, w_{i+1}, \dots, w_n , jeśli dolny poziom wieży składa się z bel o szerokościach w_i, \dots, w_j . Wtedy mamy następującą rekurencję, w której iterujemy po wszystkich możliwościach zbudowania drugiego poziomu (przyjmujemy tu oznaczenie $w[i, j] = w_i + \dots + w_j$):

$$d[i, j] = \max_{j < k \leq n} \{1 + d[j + 1, k] \mid w[i, j] \geq w[j + 1, k]\}.$$

Czas wypełniania tablicy d to $O(n^3)$, a odpowiedź to $\max_{1 \leq j \leq n} d[1, j]$.

Można nieco zmodyfikować powyższy pomysł, aby uzyskać rozwiązanie o złożoności czasowej $O(n^2)$. Oznaczmy przez $d_2[i, j]$ maksymalną wysokość wieży złożonej z bel w_i, \dots, w_n , w której dolny poziom *nie zawiera* bel w_{j+1} . Rekurencja przybierze postać

$$d_2[i, j] = \max(d_2[i, j - 1], 1 + d_2[j + 1, k_{i,j}]),$$

gdzie $k_{i,j}$ jest maksymalnym indeksem, dla którego $w[i, j] \geq w[j + 1, k_{i,j}]$. Taki indeks możemy obliczać na bieżąco, jeśli ustalimy indeks i i będziemy wypełniać tablicę d_2 kolejno dla rosnących wartości indeksu j . Odpowiedź to $d_2[1, n]$.

Jeszcze lepsze rozwiązanie uzyskamy, korzystając z obserwacji, że najwyższa wieża będzie miała też najmniejszą szerokość dolnego poziomu. Udowodnijmy to przez indukcję: pokażemy to dla wieży zbudowanej z bel w_i, \dots, w_n , przy założeniu, że teza jest spełniona dla wież zbudowanych z bel o szerokościach w_j, \dots, w_n dla $j > i$. Załóżmy, że wieża o najmniejszej szerokości dolnego poziomu ma go zbudowanego z bel w_i, \dots, w_j . Wtedy najwyższa wieża z bel w_{j+1}, \dots, w_n będzie miała (na mocy założenia indukcyjnego) największą wysokość h (a zatem cała wieża wysokość $h + 1$). Teraz dowolna inna wieża mająca dolny poziom $w_i, \dots, w_{j'}$ dla $j' > j$ ma resztę zbudowaną z $w_{j'+1}, \dots, w_n$ o wysokości h' (zatem cała wieża ma wysokość $h' + 1$). Ale wtedy istnieje wieża złożona z w_{j+1}, \dots, w_n o wysokości h' (bo wystarczy rozszerzyć dolny poziom), zatem $h' \leq h$ (z maksymalności h). To kończy dowód.

Niech zatem $d_3[i]$ oznacza taki indeks j , że najwyższa wieża zbudowana z bel o szerokościach w_i, \dots, w_n ma dolny poziom złożony z bel o szerokościach w_i, \dots, w_j . Wtedy mamy rekurencję:

$$d_3[i] = \min_{i \leq j \leq n} \{j \mid w[i, j] \geq w[j + 1, d_3[j + 1]]\}.$$

Wyznaczywszy tablicę d_3 w czasie $O(n^2)$, odpowiedź odzyskujemy, obliczając długość ciągu $d_3[1], d_3[d_3[1] + 1], \dots$ zawierającego końce kolejnych poziomów w najwyższej wieży.

Tablicę d_3 można wypełnić szybciej. Ustalmy indeks j i niech k_j będzie maksymalnym indeksem, dla którego spełniona jest nierówność $w[k_j, j] \geq w[j + 1, d_3[j + 1]]$. Wiemy zatem, że $d_3[i] \leq j$ dla wszystkich $i \leq k_j$. Algorytm jest następujący: dla kolejnych indeksów j wykonujemy przypisanie $d_3[j] := \min(d_3[j], d_3[j + 1])$, wyznaczamy wyszukiwaniem binarnym indeks k_j i wykonujemy $d_3[k_j] := j$. Złożoność czasowa tego algorytmu to $O(n \log n)$.

A Czytelników Dociekliwych zachęcamy do rozwiązania tego zadania w optymalnej złożoności czasowej $O(n)$.

Tomasz IDZIASZEK

